Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tomaž Silič

Priporočilni sistem za spletno trgovino

diplomsko delo
na univerzitetnem študiju

Mentor: prof. dr. Igor Kononenko

Ljubljana 2015

Rezultati diplomskega dela so intelektualna lastnina avtorja in Fakultete za računalništvo in informatiko Univerze v Ljubljani. Za objavljanje ali izkoriščanje rezultatov diplomskega dela je potrebno pisno soglasje avtorja, Fakultete za računalništvo in informatiko ter mentorja.

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Tomaž Silič, z vpisno številko **63080129**, sem avtor diplomskega dela z naslovom:

Priporočilni sistem za spletno trgovino

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Igorja Kononenka,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 1. junij 2015 Podpis avtorja:

Rad bi se zahvalil mentorju prof. dr. Igorju Kononenku in as. dr. Darku Pevcu za pomoč in vodenje pri izdelavi diplomske naloge. Zahvala gre tudi moji družini, ki mi je pomagala in me spodbujala tako pri doplomski nalogi, kot pri študiju.

Kazalo

Povzetek

Zaradi vse večjih potreb po hitri obdelavi velike količine podatkov se v poslovnem svetu vedno bolj pogosto uporabljajo metode podatkovnega rudarjenja znotraj različnih vrst sistemov CRM. Sistemi CRM se dopolnjujejo s tako imenovanimi priporočilnimi sistemi, ki tako strankam kot prodajalcem priporočajo izbiro raznih akcij znotraj sistema. V diplomskem delu smo pregledali obstoječe tehnike priporočilnih sistemov in posodobili manjši sistem CRM oz. spletno trgovino s priporočilnim sistemom za stranke in prodajalce. Za prodajalce smo s pomočjo algoritma Apriori tvorili povezovalna pravila med artikli glede na predhodna naročila kupcev. Povezovalna pravila so se izkazala za neuporabna, saj med artikli spletne trgovine ni bilo tesnih povezav. Za stranke pa smo po algoritmu ID3 zgradili drevo za priporočanje artiklov, ki bi utegnili stranke zanimati poleg že ogledanih artiklov. Zgradili smo dve drevesi na osnovi zgodovine obiska spletne trgovine. Podatke o obisku smo pridobili iz sistema Google Analytics.

Ključne besede

CRM, spletna trgovina, priporočilni sistemi, podatkovno rudarjenje

Abstract

Due to the increasing demand for high-speed processing of large amounts of data in the business world, data mining is becoming widely used within the different types of CRM systems. CRM systems are complemented with recommender systems that both customers and salespeople use for selecting various actions within the system. In this thesis we reviewed the existing techniques for recommendation systems and updated the CRM system i.e. an online store with a recommendation system for customers and salespeople. For salespeople we are using an algorithm Apriori which formed the association rules between products compared to previous customers' orders. Association rules have proved to be useless, because there is no close links between products. For customers, we are using ID3 algorithm to built a recommendation tree to recommend products which may be of interest to them. We have built two trees based on the history of online shop visits. Visits data were obtained from Google Analytics system.

Key words

CRM, web shop, recommendation system, data mining

Poglavje 1

Uvod

Za lažje poslovanje vsako podjetje potrebuje dober informacijski sistem, s katerim lahko obvladuje vse procese v organizaciji. Poleg tega se količina podatkov iz dneva v dan eksponentno povečuje. Uporaba priporočilnih sistemov in podatkovnega rudarjenja znotraj sistema CRM poveča učinkovitost sistema in podjetja samega, saj se s pomočjo informacij, ki jih vrača priporočilni sistem, omogoči, da se uporabniki lažje odločajo pri izbiri akcij znotraj sistema in iz podatkov dobijo želene informacije v najhitrejšem možnem času.

Priporočilni sistemi s pomočjo matematičnih in statističnih tehnik modelirajo uporabnikove preference in se skušajo priučiti njegovega profila – kaj je uporabniku všeč in kaj ne. Njihova glavna naloga je, da uporabniku na osnovi naučenega podajo čimboljše priporočilo. Dalj časa kot uporabniki uporabljajo sistem, učinkoviteje so rezultati prilagojeni njihovim preferencam.

Priporočilni sistemi postajajo vedno bolj popularni. Uporabljajo jih vse uspešnejše spletne trgovine, strani in portali, kot so npr. Amazon.com, YouTube.com, Facebook.com, itd. Spletna trgovina Amazon.com priporočilni sistem uporablja za priporočanje produktov uporabnikom na podlagi nakupov strank (npr. strake, ki so kupile ta izdelek, so kupile tudi te izdelke), na podlagi podatkov izdelka (npr. sorodni izdelki tem, ki so uporabnika že zanimali). Spletni portal YouTube.com uporabnikom priporoča video-

posnetke (npr. sorodni video-posnetki tem, ki si jih je uporabnik že ogledal). Facebook.com uporabnikom priporoča razne oglase, objave itd. Priporočilni sistemi se uporabljajo na veliko različnih področjih in so postali nepogrešljivi na vseh spletnih mestih ter v poslovanju.

Podjetje XPodjetje d.o.o. nam je zaupalo podatke njihovega sistema za vodenje strank, ki je obenem tudi njihova spletna trgovina. Ta sistem ni uporabljal metod podatkovnega rudarjenja in ni vseboval priporočilnega sistema. Nujna je bila nadgradnja sistema s priporočilnim sistemom, ki bi izboljšal uporabo spletne trgovine tako za kupce kot administratorje oz. prodajalce.

Glavni cilj diplomskega dela je torej razviti priporočilni sistem za spletno trgovino, ki bo prodajalcem pomagal pri odločanju glede popustov, stranke pa bodo hitreje našle iskani artikel. Odločili smo se za implementacijo povezovalnih pravil med artikli za pomoč prodajalcem in odločitvenega drevesa za pomoč strankam.

V prvem poglavju smo pregledali potrebno predznanje in obstoječe raziskovalne naloge iz tega področja ter jih nato skušali upoštevati pri implementaciji priporočilnega sistema. V drugem poglavju smo opisali uporabljene algoritme, tehnologije in podatke. V tretjem poglavju smo pregledali rezultate vsakega priporočilnega sistema posebej in nazadnje v četrtem poglavju komentirali dobljene rezultate ter vse skupaj na kratko povzeli.

Poglavje 2

Materiali in metode

Za doseg našega cilja smo potrebovali predznanje o priporočilnih sistemih, sistemih CRM (Customer Relationship Managment) in podatkovnem rudarjenju. Pregledali smo potrebno predznanje, tipe obstoječih priporočilnih sistemov in raziskovalne naloge iz tega področja [2, 3, 1, 5, 6, 9, 8]. S pomočjo njihovih spoznanj smo načrtovali naš priporočilni sistem in se skušali izogibati slabostim in uporabiti prednosti obstojčih podatkovnih struktur ter algoritmov. Nato smo implementirali načrtovani sistem, pri čemer smo si pomagali s podatki iz analitičnega sistema Google Analytics ter samega sistema CRM. Za stranke smo implementirali priporočilni sistem za priporočanje ogleda artiklov v primeru obiska strani nekega drugega artikla na spletni trgovini, za trgovce pa smo implementirali algoritem Apriori za povezovalna pravila med artikli v naročilih, iz katerih bi lahko pametneje določali popuste.

2.1 CRM in podatkovno rudarjenje

Customer Relationship Management (CRM) vključuje procese in sistem, ki podpira poslovno strategijo ter poskuša zagotoviti dolgoročno in donosno razmerje s strankami [2]. Podatki strank in orodja informacijske tehnologije so ključni faktorji uspešnega sistema CRM. V svetu interneta je CRM postal nepogrešljiva metoda za razvoj poslovanja, vendar zanj ne obstaja neka

splošno sprejeta definicija. Mi smo ga definirali kot sistem, ki vsebuje podatke o strankah ter razmerju med njimi in podjetjem. Te podatke analizira in interpretira na podjetju prijazen način ter s tem izboljšuje poslovanje podjetja. Tej definiciji ustreza tudi naša spletna trgovina. CRM je sestavljen iz štirih glavnih korakov:

- IDENTIFIKACIJA STRANK – pridobivanje ciljne populacije, ki bi bila najbolj zaželena in donosna. Gre za odkrivanje segmentov potencialnih strank, kjer je vsaka izmed njih relativno podobna,
- PRIVABLJANJE STRANK – organizacija poskuša direktno privabljati stranke, tako da definira proces, ki bi vzpodbudil stranko k nakupu izdelkov ali storitev podjetja skozi različne kanale. Npr. pošiljanje e-sporočil o ponudbi,
- ZADRŽEVANJE STRANK – se nanaša na aktivnosti, ki preprečujejo, da bi stranka šla h konkurenci, s povečevanjem zadovoljstva stranke z ena-na-ena marketingom, programom lojalnosti, upravljanjem pritožb,
- RAZVOJ STRANK – povečevanje števila transakcij, vrednosti transakcije in donosnosti strank. To je osnovni cilj sistema CRM.

Pristopi CRM:

- OPERATIVNI CRM – ta pristop podpira operacije in aktivnosti pri neposrednem stiku s strankami. Npr. osebje centra za sprejemanje telefonskih klicev,
- ANALITIČNI CRM – ta pristop temelji na operativnem sistemu CRM. Določa analitične informacije o segmentaciji uporabnikov, vedenje kupca in vrednost stranke,
- KOLABORATIVNI CRM – ta pristop se osredotoča na odnos povezovanja s strankami s pomočjo ustrezne komunikacijske poti (multichannel managment). Npr. integracija spletnih trgovin in centrov za sprejemanje telefonskih klicev.

2.1.1 Podatkovno rudarjenje

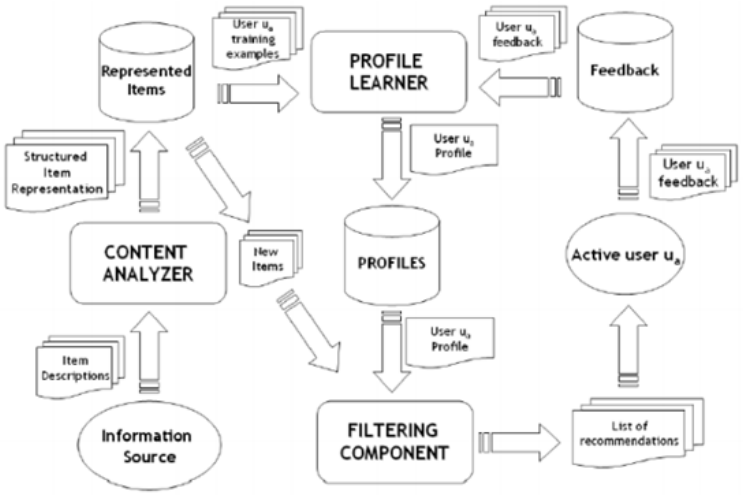
Podatkovno rudarjenje (DATA MINING) - je relativno novo znanstveno raziskovalno področje, ki je sestavljeno iz statistike, strojnega učenja in upravljanja podatkovnih baz [2]. Funkcija podatkovnega rudarjenja je razkriti znanje skrito v velikih količinah podatkov. Z razkrivanjem tega skritega znanja lahko podjetja izbirajo boljše odločitve in tako bolje poslujejo. Podatkovno rudarjenje uporablja tri glavne tehnologije:

- zbiranje in shranjevanje ogromnih količin podatkov,
- računalniki z visoko procesno močjo,
- napredni algoritmi podatkovnega rudarjenja.

Razlogi in potrebe za podatkovno rudarjenje so vse večje količine podatkov (vsakih 5 let se količina podatkov na svetu podvoji – cena shranjevanja podatkov se iz leta v leto zmanjšuje) in vse večja uporaba informacij.

2.2 Priporočilni sistemi

Priporočilni sistem je sistem, ki se s pomočjo uporabnikove zgodovine akcij priuči, kaj je uporabniku všeč oz. kaj ne, katere izdelke rad kupuje, kako pogosto kupuje, itd. [12]. Na osnovi tega nato uporabniku podaja priporočila. Več časa kot nek uporabnik uporablja sistem, bolj učinkovito mu sistem podaja priporočila, saj se z vsako interakcijo izboljšuje model uporabnika. Priporočilni sistemi so sorazmerno novo področje v primerjavi z ostalimi informacijskimi sistemi, orodji in tehnologijami. Preprosta ideja priporočilnih sistemov se je pojavila v začetku 80. let, kot računalniški program Grundy, ki je bil razvit za priporočanje knjig namesto knjižničarjev in knjižničark. V začetku 90. let pa so se, kot sodelovalno filtriranje (angl. collaborative filtering), začeli vzpenjati tudi v spletu [12, 10, 4]. Zanimanje za te sisteme se je v zadnjih letih močno povečalo. Uporabljajo jih znane internetne strani,



Slika 2.1: Visoko nivojska arhitektura vsebinsko osnovanih priporočilnih sistemov [11].

kot so YouTube, Amazon, Netflix, Yahoo, Tripadvisor, Imdb itd. Organizirajo se razne delavnice in konference, ki obravnavajo priporočilne sisteme [12]. Obravnavanje teh sistemov je prišlo tudi že v programe dodiplomskih in podiplomskih študijev [12].

2.2.1 Tipi priporočilnih sistemov

Priporočilne sisteme delimo na več tipov glede na to, koliko in katere podatke potrebujejo ter ali je potrebna interakcija uporabnika za njihovo delovanje. Poglejmo si tipe priporočilnih sistemov nekoliko podrobneje [12]:

- priporočilni sistem, osnovan na znanju – tak sistem potrebuje veliko znanja in ni primeren za področja, kjer kupci opravijo le enkraten nakup,
- priporočilni sistem na osnovi metode izbiranja s sodelovanjem – pri tej metodi uporabniki sodelujejo (izbirajo npr. artikle, ki jih zanimajo). Na podlagi tega znanja jim ponujamo npr. produkte, ki so zanimali uporabnike s podobnim zanimanjem,

- vsebinsko osnovan priporočilni sistem – pri tem tipu sistema se npr. primerjajo vsebinske lasnosti arktilov in se tako uporabniku priporočajo podobni artikli, kot so mu bili všeč v preteklosti. Na sliki 2.1 je prikazana visokonivojska arhitektura vsebinsko osnovanih priporočilnih sistemov. Proces priporočanja se izvede v treh korakih, kjer prvi korak izvaja pregledovalec vsebine (content analyzer). Ta prejme opise produktov (item descriptions), ki jih predstavi v obliki (structured item descriptions), ki je primerna za naslednji korak obdelave. V drugem koraku komponenta za učenje profila (profile learner) zbira podatke o preferencah aktivnega uporabnika (active user u_a) in na osnovi teh ustvari njegov profil. Podatke shranjuje v vir za shranjevanje odgovorov (feedback). Odzivi aktivnega uporabnika (user u_a feedback) so skupaj s sorodnimi opisi produkta uporabljeni med procesom učenja modela za napovedovanje. Priporočilni sistem se profila aktivnega uporabnika priuči s pomočjo učne množice za uporabnika u_a (user u_a training examples). Komponenta za izbiranje (filtering component) ob novi predstavitvi produkta (new items) s pomočjo profila aktivnega uporabnika določi ali produkt ustreza uporabnikovim preferencam. Ustrezajoče produkte uvrsti na seznam priporočil (list of recommendations). Uporabnik oceni produkte na seznamu priporočil, nato se ponovita prvi in drugi korak z uporabo novih spoznanj o uporabniku.

- demografski priporočilni sistem – deluje na osnovi demografskih podatkov uporabnika. Za različno demografsko nišo morajo biti podana različna priporočila,

- družbeno osnovan priporočilni sistem – uporabniku priporoča artikle na osnovi preferenc/ocen uporabnikovih prijateljev. Zbira in ureja podatke o povezavah med uporabniki,

- hibridni priporočilni sistem – osnovan na združitvi različnih metod. Sistem z združitvijo metod A in B poizkuša izkoristiti prednosti metode A, da odpravi šibkost metode B. Doseči poskuša boljša priporočila.

Od teh tipov priporočilnih sistemov nekateri (npr. priporočilni sistem na osnovi metode izbiranja s sodelovanjem) ne potrebujejo veliko znanja oz. uporabljajo zelo preproste podatke, kot so npr. uporabnikove ocene artiklov. Drugi pa so veliko bolj odvisni od znanja (npr. priporočilni sistem osnovan na znanju), zato se je pri odločanju, kateri tip priporočilnega sistema bomo implementirali, potrebno odločati tudi na osnovi znanja, ki ga imamo na voljo.

Naša spletna trgovina zbira malo podatkov, tako da sistem na osnovi znanja ni prišel v poštev. Prav tako ne zbira demografskih podatkov in podatkov o uporabnikovih prijateljih, tako sta odpadla tudi demografski in družbeno osnovan priporočilni sistem. Odločili smo se za vsebinsko osnovan sistem, ki posredno deluje tudi na osnovi metode izbiranja s sodelovanjem. Uporabniki so si ogledovali artikle (s sodelovanjem so nam posredno izdali svoje želje), mi pa smo na osnovi teh ogledov in lastnosti artiklov ustvarili priporočilni sistem. Tako je naš sistem tudi hibriden.

2.2.2 Tehnike podatkovnega rudarjenja v priporočilnih sistemih

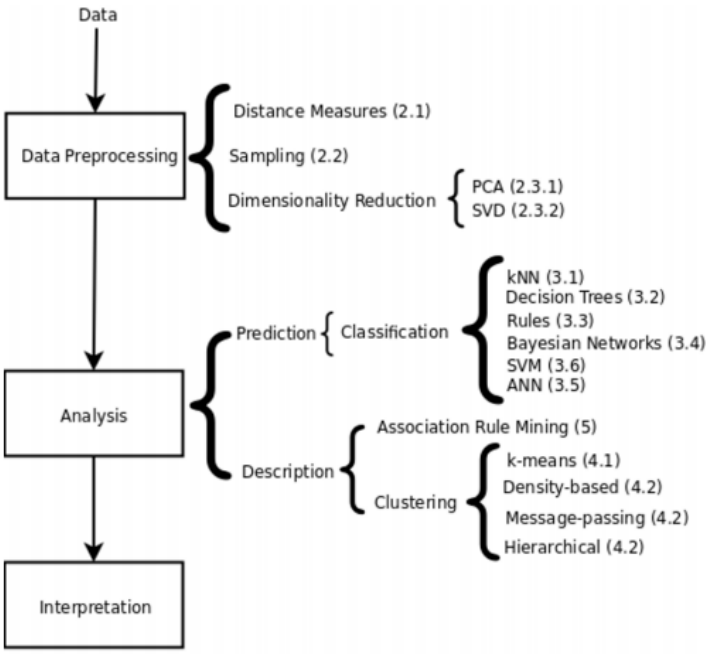
Priporočilni sistemi navadno uporabljajo tehnike iz drugih področji, kot so interakcija človek-računalnik (human computer interaction – HCI) ali pridobivanje informacij (information retrieval – IR). Vendar so ključni algoritmi, ki sestavljajo te sisteme, algoritmi podatkovnega rudarjenja.

Proces podatkovnega rudarjenja navadno sestoji iz treh korakov [7] (Slika 2.2):

1. Predpriprava podatkov (Data Preprocessing)

   Podatke definiramo kot objekte in njihove atribute, kjer so atributi definirani kot lastnost oz. karakteristika objekta.

   Realni podatki morajo biti prečiščeni, da jih lahko uporabimo za analiziranje. Lahko uporabljamo metode klasifikacije in grupiranja, ki so močno odvisne od pravilne uporabe mer za različnost oz. podobnost



Slika 2.2: Glavni koraki in metode problema podatkovnega rudarjenja [7].

podatkov. Včasih podatkovne baze vsebujejo velike količine podatkov, ki jih je težko vse obdelati v nekem realnem času, zato je potrebno ustvariti manjše vzorce, ki pa še vedno ohranjajo glavne karakteristike vseh podatkov. Nekateri podatki so difinirani v večdimenzionalnem prostoru in jim je za lažjo analizo potrebno zmanjšati število dimenzij. Velikokrat se zgodi, da podatki vsebujejo šum, kot so mankajoče vrednosti in osamelci. Zato potrebujemo tehnike za odpravo šuma v podatkih (denoising).

Tukaj se bomo omejili na merjenje različnosti oz. podobnosti podatkov, poleg tega, čeprav naša baza podatkov ni velika, si bomo pogledali tudi vzorčenje podatkov, ker ga bomo potrebovali za ustvarjanje učne in testne množice.

Različnost oz. podobnost podatkov lahko merimo z merami, kot so neenakost Minkowskega, evklidska razdalja, Mahalanobisova razdalja, kosinusna podobnost, Pearsonov korelacijski koeficient. Izmed mer različnosti se najbolj pogosto uporablja evklidska razdalja:

$$d(x, y) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2} \tag{2.1}$$

kjer sta x in y evklidska vektorja z izhodiščem v izvoru prostora, njuna konca kažeta na dve točki. n predstavlja število dimenzij. Izmed mer podobnosti pa Pearsonov korelacijski koeficient:

$$Pearson(x, y) = \frac{\sum(x, y)}{\sigma_x \times \sigma_y} \tag{2.2}$$

kjer je σ_x standardni odklon spremeljivke x in σ_y standardni odklon spremeljivke y. Z vzorčenjem podatkov lahko ustvarimo učno in testno množico. Učna množica se uporabi za učenje parametrov oz. konfiguracijo algoritmov v koraku analize podatkov, s testno množico pa preverimo konfiguracijo oz. model, pridobljen iz učne faze, glede na to, kako se odziva na še ne videne podatke.

Glavni problem vzorčenja je najti podmnožico originalne množice podatkov, ki predstavlja približno iste lastnosti, kot jih predstavlja celotna množica. Tehnike vzorčenja so:

- naključno vzorčenje – enaka verjetnost izbire kateregakoli objekta,
- stratificirano vzorčenje – podatki so razdeljeni v več particij glede na določeno značilnost, nato pa se na vsaki particiji izvede naključno vzorčenje,
- vzorčenje brez ponavljanja – ko je nek objekt v podatkih izbran, ga izključimo iz podatkov,
- vzorčenje s ponavljanjem – ko je nek objekt v podatkih izbran, ga ne izključimo iz podatkov, torej je lahko izbran več kot enkrat,
- navzkrižno preverjanje (cross-validation) – to je proces, v katerem ustvarimo učne in testne množice ter na njih izvajamo proces učenja oz. testiranja. To ponovimo K-krat, kjer je K število testnih množic. Na koncu dobimo povprečje učinkovitosti vseh naučenih K modelov.

2. Analiziranje podatkov (Data Analysis)

   Tehnik za analiziranje podatkov je veliko. Delimo jih v tri večje kategorije, kot so klasifikacija, analiza gruč in iskanje povezovalnih pravil.

   Klasifikacija je preslikava med prostorom lastnosti in prostorom oznak. Npr. spletna trgovina lahko razdeli stranke v VIP in navadne, na podlagi lastnosti, kot so število nakupov, količina denarja, porabljenega pri nakupih, itd. Poznamo vrsto metod klasifikacije, kot so: metoda najbližjih sosedov, odločitvena drevesa, Bayesovi klasifikatorji, umetne nevronske mreže, itd.

   Metode analize gruč so namenjene izboljšanju učinkovitosti priporočilnega sistema, tako da podobne objekte združimo v gruče, nato pa npr. iščemo razdalje (za iskanje k-najbližjih sosedov) med gručami namesto med objekti in tako zmanjšamo število operacij. Pri tem se nam

seveda zmanjša točnost, zato moramo najti neko pravo razmerje med izboljšanjem učinkovitosti in zmanjšanjem točnosti.

Povezovalna pravila, ki jih skušamo najti med objekti, nam napovejo verjetnost pojavitve nekega objekta v primeru pojavitve nekega drugega objekta pri transakciji (npr. pojavitev izdelka A pri pojavitvi izdelka B pri nakupu v spletni trgovini).

Naš priporočilni sistem uporablja klasifikacijo in povezovalna pravila.

3. Interpretacija rezultatov (Result Interpretation)

2.2.3 Katere značilnosti priporočilnih sistemov upoštevati pri izbiri pristopa?

Ker imajo različne aplikacije različne potrebe, se mora oblikovalec sistema odločiti o pomembnih značilnostih, kako bo oblikoval priporočilni sistem [8]:

- izbira uporabnika – uporabnikom ponudimo dva tipa sistema in od njih poizkušamo izvedeti, kateri jim je bolj všeč. Določimo uteži, ki nam povedo, izbira katerih uporabnikov nam je bolj pomembna,
- točnost napovedi – osnovna predpostavka je, da boljša priporočila kot jih bo dajal sistem, bolj bo ta všeč uporabnikom. Zato večina raziskovalcev poizkuša najti algoritem, ki vrne boljšo napoved. Točnost napovedi lahko merimo z naslednjimi merami:
  - klasifikacijska točnost,
  - cena napačne klasifikacije,
  - krivulja ROC,
- pokritost – katere artikle in/ali uporabnike upoštevati pri priporočanju. Npr. glede na popularnost ali uporabnost artiklov, izbira samo tistih uporabnikov, ki so ocenili vsaj neko določeno število artiklov, itd. Poseben primer pri pokritosti je »hladni zagon«, ko še nimamo (dovolj) podatkov o uporabnikih in artiklih,

- zaupanje sistema v napovedi – zaupanje v napovedi in priporočila sistema. Kdaj je zaupanje dovolj veliko, da sistem poda priporočilo in kdaj ne,

- zaupanje uporabnika v sistem – uporabniku prikazati tudi artikle, ki jih še pozna in so mu potencialno všeč. Na ta način uporabnik pridobi zaupanje v sistem,

- novost – filtriranje artiklov, tako da odstranimo tiste, ki jih je uporabnik že uporabil oz. ocenil. Povpraševanje uporabnika, ali je že prej poznal priporočen artikel,

- prijetna presenetljivost (angl. serendipity) – kako presenetljivo je neko pravilno priporočilo sistema. Npr. ko uporabnik dobro oceni veliko število filmov, kjer nastopa nek igralec, ni presenetljivo, da mu bo sistem priporočil nek nov film istega igralca. Naključna priporočila pa so lahko zelo presenetljiva, zato moramo najti dobro razmerje med natančnostjo in presenetljivostjo,

- raznolikost – v nekaterih primerih ni dobro uporabniku priporočati množico podobnih artiklov, saj lahko to zanj ni uporabno, ker bi trajalo dalj časa, da razišče celoten nabor artiklov. Raznolikost lahko poslabša natančnost, zato je priporočljivo računati krivulje za ocenjevanje zmanjševanja natančnosti proti izboljšanju raznolikosti,

- uporabnost – uporabnost lahko ocenjujemo npr. glede na povečanje dobička podjetja. V splošnem lahko definiramo različne tipe funkcij, ki jih poskuša priporočilni sistem optimizirati. Lahko jo definiramo kot vrednost, ki jo sistem ali pa uporabnik pridobita iz priporočila,

- tveganje – v nekaterih primerih lahko pri priporočanju pride do tveganja. Npr. pri priporočanju nakupa delnic se želijo uporabniki izogniti tveganju, da bi vrednost delnic padla. Ali pa iščejo delnice s predvideno visoko, a neogotovo rastjo,

- robustnost – to je stabilnost sistema pri prisotnosti napačnih informacij. Npr. nek lastnik hotela bi želel povečati oceno svojega hotela in bi s pomočjo lažnih profilov uporabnikov ocenjeval svoj hotel z odličnimi ocenami, ali pa bi druge hotele ocenjeval negativno. Drugi tip robustnosti pa je stabilnost sistema pri zelo veliki količini zahtev,

- zasebnost – za mnoge uporabnike je zelo pomembno, da njihove želje ostanejo zasebne, torej da nobena tretja oseba ne more uporabljati priporočilnega sistema v namen, da bi se naučila nekaj o točno določenem uporabniku,

- prilagodljivost – v primerih, kot so spletne strani z novicami, kjer je neka novica zanimiva samo za neko kratko obdobje, je zelo pomembno da je sistem hitro prilagodljiv. Taki primeri so zelo podobni kot »hladen zagon« vendar je tukaj vseeno drugače, saj lahko neka stara novica postane ponovno zanimiva,

- razširljivost – časovna in prostorska zahtevnost algoritmov. Kdaj postanejo algoritmi počasnejši? Kdaj je potrebna dodatna procesorska moč ali pa pomnilnik? Na drugi strani pa tudi, dodajanje novih postavk v sistem, kot so v našem primeru artikli in stranke.

2.2.4 Primeri priporočilnih sistemov in podatkovnega rudarjenja

Opravili smo pregled nekaterih obstoječih implementacij podatkovnega rudarjenja v sistemih CRM in nekaj primerov implementacije priporočilnih sistemov. V tem razdelku je pregledali nekaterih primerov.

1. Priporočilni sistem, ki temelji na povezovalnih pravilih

   Mican in Tomai [5] sta leta 2010 zgradila modularni sistem za personalizacijo in prilagodljive spletne aplikacije, ki je temeljil na povezovalnih pravilih. Predlagala sta uporabo pametnega priporočilnega sistema

(WRS – Wise Recommender System), kjer se podatki pridobivajo implicitno, brez potrebe po interakciji uporabnika. Podatki, pridobljeni iz takega sistema, so kvalitetni, popolni, brez šuma in brez napak.

Podatke sta pridobivala iz vsake seje uporabnika. Ugotavljala sta, do katerih strani uporabnik dostopa oz. natančen potek njegove seje na spletni strani in vse te podatke zapisovala v tabelo o uporabnikovi seji. Modul za generiranje povezovalnih pravil lahko dostopa do podatkov in iz njih generira povezovalna pravila v realnem času. Kadar modul obdela neko sejo, se podatki izbrišejo iz tabele. Tako sta dosegla razširljiv model, ki lahko dela v realnem času z veliko količino podatkov.

Tak modul je zelo natančen pri priporočanju vsebine in je z lahkoto implementiran v katerokoli spletno aplikacijo.

2. Priporočilni sistem, ki temelji na odločitvenih drevesih

   Glavna slabost uporabe odločitvenih dreves za napovedovanje v priporočilnih sistemih je potreba po izgradnji velikega števila dreves. Poleg tega je potrebno ta drevesa prečkati od korena do lista za vsak objekt posebej.

   Gershman in sodelavci [6] predlagajo prilagoditev odločitvenih dreves, tako da so ta primernejša za priporočilne sisteme z veliko količino podatkov. Namesto napovedovanja ocene nekega objekta drevo vrne utežen seznam priporočenih objektov, tako lahko samo z enim prehodom od korena do lista zgradimo napoved in jo podamo uporabniku. Poleg tega so podali nov kriterij hevristike za gradnjo odločitvenih dreves.

   Drevesa so zgradili tako, da so združili drevesa, pridobljena z metodo izbiranja s sodelovanjem, ter vsebinsko osnovana drevesa v eno samo hibridno drevo. Atributi, po katerih se dogaja razvejanje drevesa, so samo atributi, ki opisujejo uporabnika. Največja prednost teh dreves so listi, v katerih se nahaja seznam priporočil sistema.

Namesto popularnega algoritma C4.5 za določitev vejanja v vozliščih so uporabili lastno hevristiko, ki se je izkazala za veliko boljšo. Ta hevristika predlaga, da je boljše tisto vejanje, kjer je presek med podmnožico A in podmnožico B, ki ju dobimo iz vejanja, manjši.

3. Priporočilni sistem za priporočanje filmov, ki temelji na SVM in algoritmu K-means

   Eyjolfsdottir in sodelavci [9] so razvili priporočilni sistem, specializiran za priporočanje filmov. Pri tem so uporabili SVM (Support vector machines) za napoved žanra in dolžine filma, ki si ga nek uporabnik najbolj želi. Filme so nato razporedili v gruče s pomočjo algoritma K-means in s pomočjo teh gruč je algoritem ustvaril vprašanja za uporabnike, s katerimi so nato izboljšali napoved.

Poglavje 3

Implementacija

Opravili smo nadgradnjo spletne trgovine s priporočilnim sistemom za stranke in prodajalce. Priporočilni sistem za stranke s pomočjo podatkov iz analitičnega sistema Google Analytics in podatkovne baze spletne trgovine gradi drevo, s pomočjo katerega se priuči, kateri artikli so na osnovi obiskane strani spletne trgovine zanimivi za določen tip uporabnika. Priporočilni sistem za prodajalce pa na osnovi algoritma Apriori gradi povezovalna pravila med artikli in poizkuša ugotoviti, kakšne so povezave med njimi. V sledečih razdelkih so opisane uporabljene tehnologije, obdelani podatki in implementacija algoritmov.

3.1 Uporabljene tehnologije

Spletna trgovina je postavljena na osnovi v spletu pogosto uporabljenih tehnologij, kot so: PHP, MySql, HTML, Javascript ter CSS. Za nadgradnjo smo od teh tehnologij v veliki meri uporabljali PHP, za povezavo z analitičnim sistemom Google Analytics pa smo morali opraviti tudi manjše spremembe v jeziku Javascript. Spletna trgovina kot analitično orodje uporablja Google Analytics, s katerim smo se povezali preko aplikacijskega vmesnika oz. API (Application programming interface) in tako, poleg podatkov iz podatkovne baze, pridobili dodatne podatke za obdelavo.

3.1.1 PHP, MySQL in Javascript

PHP (Hypertext Preprocessor oz. izvirno Personal Home Page Tools) je spletni programski jezik, ki je v spletu pogosto uporabljen [16]. Oblikovan je bil za programiranje na strani strežnika, kar pomeni, da se vse operacije izvajajo na strežniku. PHP ima tudi sposobnost prepletanja s HTML, tako je mogoče združiti obliko s programiranjem.

MySQL je odprtokodni sistem za upravljanje s podatkovnimi bazami. Kot je že iz imena razvidno, uporablja jezik SQL (Structured Query Language), strukturirani povpraševalni jezik za delo s podatkovnimi bazami [15]. Ta jezik je najbolj razširjen povpraševalni jezik za delo s podatkovnimi zbirkami.

Javascript je programski jezik, ki se za razliko od PHP izvaja na strani klienta [14]. Google Analytics ga izrablja za nastavljanje piškotkov ter pošiljanje podatkov o obisku strani.

3.1.2 Google Analytics in aplikacijski vmesnik

Spletna storitev Google Analytics omogoča pridobivanje, hranjenje in pregled podatkov vseh uporabnikov na našem spletnem mestu [13]. Za pridobivanje podatkov uporablja programski jezik Javascript, s pomočjo katerega nastavi piškotke za ločevanje sej in uporabnikov ter te podatke pošilja v podatkovno bazo sistema.

Aplikacijski vmesnik je množica rutin, protokolov in orodij za gradnjo programskih aplikacij. Vmesnik določa, kakšna naj bo interakcija med programskimi komponentami. Aplikacijski vmesnik sistema Google Analytics sestoji iz več vmesnikov, ki so namenjeni različnim funkcijam. Mi smo uporabili vmesnik s funkcijami za povpraševanje po dimenzijah in meritvah (Core Reporting API).

3.2 Podatki

Kot smo že omenili, smo podatke pridobivali preko aplikacijskega vmesnika iz storitve Google Analytics in podatkovne baze spletne trgovine. Podatke iz storitve Google Analytics smo uporabili za gradnjo drevesa in napovedovanje priporočenih artiklov. Iz podatkovne baze pa smo uporabili podatke o naročilih in njihovih artiklih za tvorjenje povezovalnih pravil z algoritmom Apriori.

3.2.1 Podatki iz storitve Google Analytics

Storitev Google Analytics privzeto omogoča pregled obiska strani po številu sej oz. uporabnikov na nek dogodek. Za potrebe našega priporočilnega sistema pa smo morali poznati vrstni red dogodkov za posameznega uporabnika. Npr. če je nek uporabnik obiskal stran z artiklom A, nas je zanimalo, kateri artikli so ga zanimali poleg artikla A (strani katerih artiklov je obiskoval v isti seji). Za pridobitev teh podatkov je bila potrebna manjša nadgradnja v programski kodi za sledenje uporabnikov. Google Analytics v piškotih hrani identifikator seje. Ker smo želeli spremljati vsako sejo posebej, smo morali ta identifikator, poleg privzetih podatkov, kot razežnost po meri poslati statističnemu sistemu. Ker je bilo potrebno to nadgradnjo še implementirati, je bila količina podatkov, ki smo jih obdelovali, veliko manjša, saj so bili podatki pred nadgradnjo neuporabni za poizvedbe, ki smo jih želeli opraviti.

Google Analytics hrani različne razsežnosti in spremenljivke (v nadaljevanju atributi). Za naše potrebe so bile zanimive: identifikator seje, država, obiskan url-naslov, tip uporabnika (obstoječ ali nov), spol in starost. Spola in starosti žal ni bilo mogoče spremljati, ker storitev ne dovoljuje vpogleda v demografske podatke v primeru, da je število sej manjše od nekega števila N. V našem primeru je bilo število sej na niz izbranih atributov enako 1, saj nas je zanimala posamezna seja. Tako smo izbrali diskretne atribute država, obiskan url-naslov, tip uporabnika in identifikator seje.

Država	Tip uporabnika	Url naslov	ID Seje
Slovenia	Returning Visitor	/product/74	700404180.1429864786
Germany	New Visitor	/product/79	700404180.1429864786
Italy	Returning Visitor	/product/13	700404180.1429864786
Slovenia	New Visitor	/product/70	700404180.1429864786

Tabela 3.1: Primeri še neobdelanih podatkov.

Za zbiranje in obdelavo podatkov smo v jeziku PHP implementirali dva razreda. Razred za povezavo s storitvijo Google Analytics, ki vsebuje tri funkcije za povezavo s storitvijo in avtentikacijo in pridobivanje želenih podatkov. Poleg tega smo implementirali še razred za opis podatkov, ki vsebuje vse potrebne funkcije za opis in predhodno obdelavo podatkov. Pridobljene podatke smo predhodno obdelali tako, da smo iz celotne množice podatkov odstranili tiste podmnožice atributov, ki niso vsebovale naslovov strani z artikli, saj nas obisk le-teh ni zanimal. Iz url-naslovov smo izluščili identifikator artikla in atribut url-naslov nadomestili s tem identifikatorjem. Glede na identifikator seje smo vsaki množici atributov dodali ciljni atribut oz. razred, identifikator artikla, ki je bil obiskana v isti seji kot trenutno obiskana stran. Ker je identifikator seje služil samo dodajanju ciljnega atributa, smo ga odstranili iz množice atributov pred gradnjo drevesa. Nazadnje smo atribut država nadomestili z binarnim atributom, ki nam je povedal, ali je kupec Slovenec ali tujec. V tabeli 3.1 je prikazanih nekaj primerov še ne obdelanih podatkov, v tabeli 3.2 pa nekaj primerov predhodno že obdelanih istih podatkov.

3.2.2 Podatki iz podatkovne baze

Podatkovna baza sistema oz. spletne trgovine uporablja jezik MySql, omenjen v razdelku uporabljenih tehnologij. V podatkovni bazi je veliko tabel, ki jih spletna trgovina potrebuje za svoje delovanje.

Država	Tip uporabnika	ID Artikla	ID Seje	ID Artikla 2
Slovenia	Returning Visitor	74	700404180.1429864786	13
Tujina	New Visitor	79	700404180.1429864786	88
Tujina	Returning Visitor	13	700404180.1429864786	65
Slovenia	New Visitor	70	700404180.1429864786	69

Tabela 3.2: Primeri predhodno obdelanih podatkov.

Za algoritem Apriori sta bili zanimivi tabela naročil in tabela artiklov na naročilih. Tabelo naročil sestavljajo vsi podatki kupca in prejemnika v času naročila, skupna cena naročila, davek, uporabljen kupon, popusti itd. Tabelo artiklov na naročilih pa sestavljajo vsi podatki artiklov v času naročila. Tabelo naročil bi potrebovali za podrobnejše raziskave o nakupih, ki jih nato nismo izvajali, saj smo ugotovili, da med artikli v naši spletni trgovini ni tesnih povezav in tako se je priporočilni sistem izkazal za neuporabnega. Tabelo artiklov pa smo potrebovali za tvorjenje povezovalnih pravil. Algoritem je s poizvedbo iz tabele potegnil vsa imena artiklov na naročilih ter tuji ključ, ki artikel povezuje z naročilom, ter iz teh podatkov sestavil povezovalna pravila.

Za algoritem ID3 pa je bila zanimiva tabela vseh artiklov v spletni trgovini. Iz te tabele smo uporabili tri atribute, ki natančneje opisujejo večino artiklov, in jih uporabili za en primer gradnje drevesa. Delež artiklov (približno 20 %) teh atributov ne uporablja.

3.3 Opis algoritmov

Za gradnjo drevesa smo uporabili algoritem ID3 ter algoritem Apriori za tvorjenje povezovalnih pravil. Algoritem za gradnjo drevesa smo nadgradili, tako da je namesto enega artikla v listih vračal seznam petih najbolje uteženih artiklov, in tako zmanjšali število prehodov preko drevesa pri napovedovanju. Pri algoritmu Apriori so rezultati, ki smo jih obdelali v četrtem poglavju, pokazali, da je nadaljnje delo na povezovalnih pravilih brezpredmetno, saj med artikli v naši spletni trgovini ni tesnih povezav, zato algoritma nismo

poskušali nadgrajevati.

3.3.1 Gradnja drevesa po algoritmu ID3

Algoritem ID3 je klasifikacijski algoritem, ki iz končne množice podatkov zgradi drevo. Dobljeno drevo je nato uporabljeno za razvrščanje prihodnjih primerov. Vsak primer ima več atributov, ki pripadajo nekemu razredu. Listi odločitvenega drevesa vsebujejo ime razreda oz. v našem primeru imena večih razredov, ostala vozlišča pa so odločitvena vozlišča [17].

Zgradili smo dve drevesi iz različnih atributov ter primerjali, katero drevo boljše napoveduje zanimanje uporabnikov. Imeli smo torej dve množici podatkov. Prva je vsebovala štiri atribute, opisane v prejšnjem razdelku: ID seje, država, obiskan url-naslov ter tip uporabnika (obstoječ ali nov). Kot že omenjeno, smo podatke predhodno obdelali in za vsak primer dobili naslednjo množico atributov:

- TIP UPORABNIKA – Atribut pove, ali je uporabnik že kdaj obiskoval našo spletno tgovino ali ne,
- IDENTIFIKATOR TRENUTNO OBISKANEGA ARTIKLA – Identifikator artikla, na katerem se je uporabnik trenutno nahajal,
- TUJINA – Binarni atribut, ki nam je povedal, ali je kupec tujec ali Slovenec.

Druga množica je tako kot prva vsebovala atributa “tip uporabnika” ter “tujina”, identifikator artikla pa smo nadomestili z lastnostmi artikla iz podatkovne baze.

Poleg teh atributov smo vsakemu primeru iz obeh množic dodali še razred “identifikator drugega artikla”, ki je označeval artikel, obiskan v isti seji. Razred smo dodali s pomočjo zanke, v kateri smo se sprehodili čez celotne podatke in z uporabo atributa “identifikator seje” določili, kateri url-naslovi so bili obiskani v isti seji kot trenutno obiskani naslov. Nazadnje smo atribut “identifikator seje” odstranili iz množice atributov za klasifikacijo.

Agoritem ID3 je izboljšava algoritma CLS (Concept Learning System) z uporabo hevristike [17]. Algoritem preišče vse atribute in se s pomočjo hevristike odloči, kateri najbolje razdeli množico podatkov. Za preiskovanje uporablja požrešno preiskovanje (greedy search), kar pomeni, da izbere najboljši atribut in nikoli ne gleda nazaj, če bi bilo potrebno ponovno pretehtati prejšnje izbire. Za izračun, kateri atribut najbolje razdeli množico podatkov v danem vozlišču algoritem ID3 uporablja razmerje informacijskega prispeveka. Atribut z največjo informacijo je izbran za vejanje. Za izračun razmerja informacijskega prispevka smo potrebovali še mero entropije, ki meri količino informacije nekega atributa.

Naš objekt za izgradnjo drevesa, ki je priložen v dodatku C, vsebuje funkcijo, ki s pomočjo rekurzije gradi drevo. Ta funkcija vsebuje tudi objekt s funkcijami za izračun entropije (dodatek B) in razmerja informacijskega prispevka ter objekt vozlišče za dodajanje vozlišč v drevo. Algoritem najprej izbere atribut, ki bo v korenu drevesa. Izračuna se entropija vsakega izmed atributov po enačbi 3.1, kjer je p_i delež S iz razreda i, S pa predstavlja atribut.

$$E(S) = \sum_{i=1}^{c} -p_i log_2 p_i \tag{3.1}$$

Nato se v zanki sprehodimo čez podatke in za vsak atribut se izračunajo pogojne entropije atributa X glede na vrednosti razreda T po enačbi 3.2, kjer je c vrednost atributa X, $P(c)$ verjetnost, da bo atribut X imel vrednost c in $E(c)$ entropija vrednosti c pri razredu T.

$$E(T, X) = \sum_{c \in X} P(c)E(c) \tag{3.2}$$

Nato se izračuna razmerje informacijskega prispevka za dani atribut kot razlika med entropijo ciljnega atributa in vsoto pogojnih entropij danega atributa glede na vrednosti atributa po enačba 3.3, kjer je $\sum$ vsaka vrednost v vseh možnih vrednosti atributa A, S_v podmnožica S, za katero ima atribut A vrednost v, $|S_v|$ število elementov v S_v in $|S|$ število elementov v S.

$$Gain(S, A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v) \tag{3.3}$$

Atribut z največjim razmerjem se izbere za vejanje. Nato se pomakne v naslednje vozlišče in ponovi postopek. Ta proces se ponavlja, dokler niso klasificirani vsi podatki ali pa nam zmanjka atributov. V tem razredu je še funkcija, ki ustvari list drevsa, ko pride do razreda.

Algoritem v listu drevesa nato vrne najbolje uteženo vrednost iz razreda. Mi smo ta algoritem spremenili, tako da v listih vrača seznam petih najbolje uteženih vrednosti, in s tem zmanjšali prehode čez drevo pri napovedovanju. Zgrajeno drevo se nazadnje shrani v datoteko JSON, da ga lahko uporabimo pri napovedovanju.

Pri napovedovanju algoritem poiskuša primer z atributi umestiti v drevo. Kot rezultat napovedi vrne list drevesa s petimi najbolje uteženimi artikli.

3.3.2 Tvorjenje povezovalnih pravil z algoritmom Apriori

Povezovalna pravila so ena izmed najbolj pomembnih funkcionalnosti podatkovnega rudarjenja. Z njimi poskušamo najti neznane povezave, ki tvorijo osnovo za odločanje in napovedovanje. Za tvorjenje povezovalnih pravil smo uporabili algoritem Apriori [1]. Ta algoritem je enostaven za izvedbo in zelo preprost, uporablja se za rudarjenje pogostih podmnožic v podatkovni bazi. Algoritem opravi več iskanj v podatkovni bazi, da bi našel pogoste podmnožice, kjer so podmnožice velikosti k uporabljene za tvorjenje podmnožic velikosti $k + 1$. Vsaka podmnožica se mora v bazi pojaviti vsaj tolikokrat, kolikor je minimalna podpora. Algoritem najprej opravi pregled čez celotno bazo artiklov, ki so bili vsebovani v naročilih, in prešteje, kolikokrat se je kateri izmed njih pojavil. Nato so vsi artikli, ki so bili vsebovani vsaj tolikokrat, kolikor je bila minimalna podpora, uporabljeni za tvorjenje podmnožic dveh artiklov. Podmnožice dveh artiklov so nato uporabljene za tvorjenje podmnožic s tremi artikli in tako dalje, dokler ni več podmnožic neke dosežene velikosti k. Če katera izmed podmnožic ne dosega minimalne podpore, tudi večje podmnožice, generirane iz nje, ne bodo dosegale minimalne podpore. S pomočjo tega pogoja se iskalni prostor algoritma zmanjša in tako je algoritem

hitrejši.

Iz vsake podmnožice velikosti k so nato generirana povezovalna pravila. Npr. iz podmnožice $X = \{A1, A2, A3\}$ algoritem ustvari povezovalna pravila, kot so $\{A1 \rightarrow A2, A3\}$, $\{A2 \rightarrow A1, A3\}$, $\{A3 \rightarrow A1, A2\}$, $\{A1, A2 \rightarrow A3\}$, $\{A1, A3 \rightarrow A2\}$, $\{A2, A3 \rightarrow A1\}$, število teh pravil je $n^2 - 1$, kjer je n število artiklov. Za potrditev povezovalnega pravila algoritem uporablja mere podpora in zaupanje. Podpora $supp(X)$ pove, v koliko odstotkih naročil se je pojavila neka podmnožica artiklov X. Zaupanje za povezovalno pravilo $X \implies Y$ pa pove, koliko odstotkov je naročil, ki vsebujejo X in Y od teh, ki vsebujejo samo X (Enačba 3.4). Minimum obeh mer je definiran s strani uporabnika in predstavlja omejitev za tvorjenje povezovalnih pravil.

$$conf(X \implies Y) = \frac{supp(X \cup Y)}{supp(X)} \tag{3.4}$$

Algoritem Apriori, uporabljen v tem diplomskem delu, je nadgrajena verzija, ki sta jo predlagala Al-Maolegi in Arkok [1]. Algoritem se samo enkrat sprehodi čez bazo naročil in tvori L_1, ki vsebuje artikle, njihovo podporo in ID transakcije, kjer je bil artikel najden. $L_2, L_3, ...L_k$ so nato pridobljeni s pomočjo L_1. Implementacija objekta za tvorjenje povezovaljnih pravil po algoritmu Apriori je priložena v dodatku A.

Poglavje 4

Rezultati

Pri pregledu rezultatov je bila naša prva ugotovitev, da spletna trgovina zbira veliko premalo podatkov za obdelavo, da bi lahko iz nje dobili željene analize in iz njih tvorili dober priporočilni sistem. Priporočilni sistem, ki smo ga razvili za prodajalce, se je izkazal za neuporabnega, saj povezovalnih pravil med artikli praktično ni. Priporočilni sistem za stranke pa bi potreboval nadgradnje sistema pri zbiranju podatkov. Poglejmo si rezultate za vsak priporočilni sistem posebej.

4.1 Priporočilni sistem za prodajalce

Priporočilni sistem za prodajalce je poskušal tvoriti povezovalna pravila, ki bi prodajalcem pomagala izboljšati odločanje pri postavitvah popustov in tvorjenju paketov iz tesno povezanih artiklov. Iz dobljenih rezultatov smo ugotovili, da je v primeru naše spletne trgovine tak sistem neuporaben, saj se v večini primerov artikli kupujejo posamezno in med njimi ni tesnih povezav.

Algoritem smo pognali na 727 naročilih, ki so skupaj vsebovala 82 različnih artiklov. Algoritem smo pognali večkrat, začenši z minimalno podporo 0.01, kar pomeni, da je morala biti generirana podmnožica vsebovana v 1 % vseh naročil. Minimalno podporo smo nato povečevali in že pri minimalni podpori 0.03 algoritem ni več tvoril podmnožic velikosti 2, kar pomeni, da je bila tudi

množica povezovalnih pravil prazna. Minimalno podporo smo uspeli povečati do 0.026, minimalno zaupanje pa do 0.4. V tem primeru je algoritem tvoril samo še eno povezovalno pravilo.

4.2 Priporočilni sistem za stranke

Kot pravijo Gershman in sodelavci [6], so lahko odločitvena drevesa uporabljena v različnih pristopih priporočilnih sistemov:

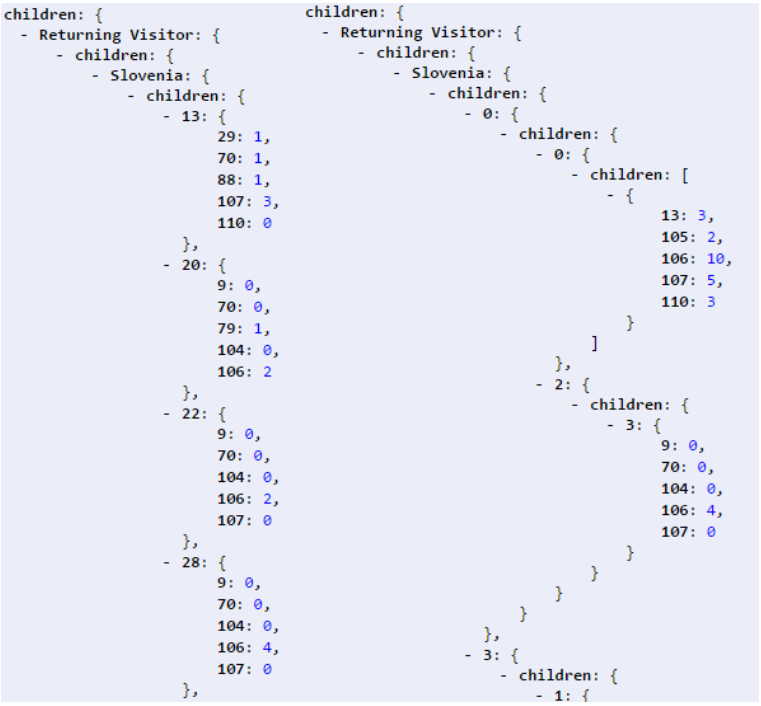
- SODELOVALNO FILTRIRANJE (Collaborative Filtering) – V tem primeru se za vsak element v sistemu (v našem primeru so to artikli) zgradi lastno drevo na osnovi uporabnikove povratne informacije,
- VSEBINSKI PRISTOP (Content-Based Approach) – V tem primeru se zgradi drevo za vsakega uporabnika, kjer so lastnosti vsakega elementa uporabljene za izgradnjo modela, ki razloži uporabnikove želje,
- HIBRIDNI PRISTOP (Hybrid Approach) – Hibridni pristop je združitev ostalih dveh, kjer zgradimo samo eno drevo, ki je podobno tistemu iz sodelovalnega filtriranja, vendar za vejanje uporablja uporabnikove atribute.

Mi smo uporabili Hibridni pristop, s katerim smo ustvarili dve drevesi.

4.2.1 Opis zgrajenih dreves

Prvo drevo sestoji iz največ treh nivojev, odvisno na katerem nivoju je bila že mogoča odločitev. V korenu se drevo razveji glede na tip uporabnika. Na vozliščih drugega nivoja se razveji glede na atribut "tujina", na tretjem nivoju pa glede na identifikator obiskanega artikla.

Drugo drevo, pridobljeno delno iz atributov iz sistema Google Analytics in delno iz podatkovne baze, sestoji iz največ pet nivojev. V korenu se drevo tako kot prvo razveji glede na tip uporabnika, nato glede na atribut "tujina", na zadnjih treh nivojih pa glede na tri lastnosti artiklov. Obe drevesi v listih



Slika 4.1: Del zgrajenih dreves v obliki JSON. Desno je drevo z atributi iz podatkovne baze in sistema Google Analytics.

vrneta seznam identifikatorjev petih najbolj obiskanih artiklov, ki so bili obiskani v isti seji kot obiskan artikel primera.

Slika 4.1 prikazuje primer dela obeh dreves v obliki JSON. Levo drevo je naše prvo drevo, ki sestoji iz največ treh nivojev. Na sliki so vidni vsi trije nivoji. Vsak nivo predstavlja ključ "children", ki označuje vse otroke vozlišča. Iz primera je razvidno, da bodo npr. obiskovalca iz Slovenije, ki je spletno trgovino že obiskoval, ob obisku artikla z identifikatorjem 13, najverjetneje zanimali artikli z identifikatorji 29, 70, 88, 107 in 110. V listih je poleg identifikatorja artikla še utež, ki označuje, kolikokrat se je zgodil ta primer. Desno drevo je naše drugo drevo, ki poleg atributov iz sistema Google Analytics vsebuje še tri atribute iz podatkovne baze. Za razliko od levega drevesa ima to drevo maksimalno globino 5. Iz tega primera je razvidno, da bodo npr. obiskovalca iz Slovenije, ki je spletno trgovino že obiskoval, ob obisku artikla, ki ne uporablja lastnosti (identifikator lastnosti 0 pomeni, da artikel nima izbrane lastnosti), zanimali artikli z identifikatorji 13, 105, 106, 107 in 110.

4.2.2 Napovedovanje

Množica primerov, ki smo jih dobili za gradnjo odločitvenega drevesa, je bila odločno premajhna za gradnjo dobrega drevesa. Največji razlog za tako malo podatkov je implementacija, omenjena v razdelku 3.2.1. Poleg tega je bilo potrebno množico primerov že v procesu predobdelave podatkov še dodatno zmanjšati, saj je vsebovala veliko podatkov, ki nas niso zanimali (npr. obisk strani, ki ne vsebujejo artiklov). Tako smo dobili 4902 primera, iz katerih je algoritem sestavil obe drevesi. Primer pri prvem drevesu je vseboval 3 atribute in 1 razred, primer pri drugem drevesu pa je vseboval 5 atributov in 1 razred. Vsi atributi in razred so bili diskretni. Množico teh primerov smo razdelili na učno in testno po postopku k-kratnega prečnega preverjanja, kjer k označuje število podmnožic, v katere razdelimo primere. Nato drevo zgradimo k-krat, tako da vsakič izberemo drugo podmnožico za testno in vse ostale združimo v učno. Postopek smo ponovili dvakrat in sicer z 10-kratnim

Testna podmnožica	Uspešnost 1. drevesa	Uspešnost 2. drevesa
1.	13.65 %	25.87 %
2.	9.75 %	21.59 %
3.	27.29 %	28.72 %
4.	22.81 %	24.44 %
5.	14.66 %	16.5 %
6.	23.42 %	21.38%
7.	24.85 %	28.92 %
8.	16.09 %	23.63 %
9.	13.85 %	22.4 %
10.	22.77 %	23.6 %
Povprečno	18.9 %	23.7 %

Tabela 4.1: Tabela uspešnosti odločitvenih dreves pri 10-kratnem prečnem preverjanju.

in 5-kratnim prečnim preverjanjem.

Ker bomo strankam prikazali vseh 5 artiklov med priporočenimi artikli, smo pri preverjanju pravilnosti napovedi upoštevali za pravilno vsako napoved, kjer je bil napovedani artikel eden izmed petih priporočenih.

Rezultati 10-kratnega prečnega preverjanja so prikazani v tabeli 4.1, rezultati 5-kratnega prečnega preverjanja pa v tabeli 4.2.

Pri 10-kratnem prečnem preverjanju je bila povprečna uspešnost prvega drevesa 18.9 %, drugega pa 23.7 %. Pri 5-kratnem prečnem preverjanju je bila uspešnost prvega drevesa 17.91 %, drugega pa 20.14 %. Obe preverjanji kažeta na to, da je drugo drevo nekoliko bolj uspešno kot prvo, vendar je ta razlika veliko premajhna, da bi se lahko odločili, katero je boljše. Da bi zgradili dobro drevo, bi potrebovali veliko več podatkov, kot jih je bilo na voljo. Poleg tega bi bilo potrebno obe drevesi preizkusiti še v praksi, saj bi se najverjetneje odzivali drugače in takrat bi lahko bilo prvo drevo boljše od drugega.

Testna podmnožica	Uspešnost 1. drevesa	Uspešnost 2. drevesa
1.	6.32 %	8.15 %
2.	25.18 %	24.46 %
3.	19.06 %	18.35 %
4.	20.18 %	26.3 %
5.	18.81 %	23.42 %
Povprečno	17.91 %	20.14 %

Tabela 4.2: Tabela uspešnosti odločitvenih dreves pri 5-kratnem prečnem preverjanju.

Poglavje 5

Razprava in sklepne ugotovitve

Cilj diplomskega dela je bil implementirati priporočilni sistem za spletno trgovino za kupce in prodajalce. V prvem delu smo pregledali potrebno predznanje in obstoječe raziskovalne naloge iz tega področja. Pri implementaciji smo uporabili pridobljeno znanje in upoštevali prednosti ter slabosti raznih algoritmov. Na podlagi pridobljenega znanja in podatkov, ki so nam bili na voljo, smo se odločili implementirati priporočilni sistem za prodajalce, ki bi jim na osnovi povezovalnih pravil skušal priporočati, katere artikle naj združujejo v skupine artiklov, na katere lahko ob nakupu stranka dobi popust. Povezovalna pravila smo tvorili s pomočjo algoritma Apriori. Nazadnje smo implementirali še priporočilni sistem za stranke, ki je deloval na osnovi algoritma ID3 za gradnjo odločitvenih dreves. Priporočilni sistem je namenjen priporočanju artiklov strankam, za katere meni, da bi jih utegnili zanimati. Kateri artikli bi stranke zanimali, se sistem odloči na osnovi ogledov strani predhodnih uporabnikov.

Za priporočilni sistem za prodajalce smo ugotovili, da je neuporaben za dano spletno trgovino, saj med artikli ni tesnih povezav. Artikli se večinoma kupujejo posamezno. Za prodajalce bi tako bilo potrebno razviti drugačen priporočilni sistem, za katerega pa spletna trgovina zbira premalo podatkov.

Isti problem s premajhno količino podatkov se pojavlja tudi pri priporočilnem sistemu za stranke, saj nam zaradi tega ni uspelo vzpostaviti

dobrega priporočilnega sistema. Za razliko od priporočilnega sistema za prodajalce je ta uporaben, vendar je potrebno počakati na večjo količino podatkov in zgraditi boljše odločitveno drevo.

Spletna trgovina je s priporočilnim sistemom za stranke sicer pridobila na svoji vrednosti, vendar ji do dobrega sistema CRM manjka še veliko. Glavna nadgradnja, ki je v prihodnosti potrebna, je pridobivanje in hranjenje čimvečje količine podatkov, iz katere lahko potem nastane tudi dober priporočilni sistem.

Literatura

[1] M. Al-Maolegi, B. Arkok, "An Improved Apriori Algorithm for Association Rules", International Journal on Natural Language Computing (IJNLC), št. 1, Februar 2014

[2] M. B. Nejad, E. B. Nejad, A. Karami, "Using Data Mining Techniques to Increase Efficiency of Customer Relationship Management Process", Research Journal of Applied Sciences, Engineering and Technology 4, št. 23, 2012, str. 5010–5015

[3] E. W. T. Ngai, L. Xiu, D. C. K. Chau, "Application of data mining techniques in customer relationship management: A literature review and classification", Expert Systems with Applications, št. 36, 2009, str. 2592–2602

[4] E. Rich, "User Modeling via Stereotypes", Cognitive Science, št. 3, 1979, str. 329–354

[5] D. Mican, N. Tomai "Association-Rules-Based Recommender System for Personalization in Adaptive Web-Based Applications", Proceedings of the 10th international conference on Current trends in web engineering, 2010, str. 85-90

[6] A. Gershman, A. Meisels, K. Lüke, L. Rokach, A. Schclar, A. Sturm, "A Decision Tree Based Recommender System", IICS, 2010, str. 170-179

[7] X. Amatriain, A. Jaimes, N. N. Oliver, J. M. Pujol, "Data Mining Methods for Recommender Systems", Springer, 2011, str. 39-71

[8] G. Shani, A. Gunawardana, "Evaluating Recommendation Systems", Microsoft Research, 2009

[9] E. A. Eyjolfsdottir, G. Tilak, N. Li, "MovieGEN: A Movie Recommendation System", Technical Report

[10] M. D. Ekstrand, J. T. Riedl, J. A. Konstan "Collaborative Filtering Recommender Systems", Human-Computer Interaction Vol 4., Št. 2, 2010, str. 81-173

[11] F. Ricci, L. Rokach, B. Shapira, P. B. Kantor "Recommender Systems Handbook", Springer, 2010

[12] B. Bahar, "Primerjava različnih tipov priporočilnih sistemov", Diplomsko delo, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, Ljubljana, 2012

[13] Google Analytics (September 2015). Dostopno na: https://en.wikipedia.org/wiki/Google_Analytics

[14] JavaScript (September 2015). Dostopno na: https://en.wikipedia.org/wiki/JavaScript

[15] MySQL (September 2015). Dostopno na: https://en.wikipedia.org/wiki/MySQL

[16] PHP (September 2015). Dostopno na: https://en.wikipedia.org/wiki/PHP

[17] D. Dankel, The ID3 Algorithm (September 2015). Dostopno na: http://www.cise.ufl.edu/ ddd/cap6635/Fall-97/Short-papers/2.htm